IN THE UNITED STATES DISTRICT COURT
FOR THE WESTERN DISTRICT OF TEXAS
MIDLAND-ODESSA DIVISION

MALIKIE INNOVATIONS LTD.,
KEY PATENT INNOVATIONS LTD.

Plaintiffs,

v.

MARA HOLDINGS, INC. (F/K/A
MARATHON DIGITAL HOLDINGS, INC.)

Defendant.

CASE NO. 7:25-CV-00222-DC-DTG

JURY TRIAL DEMANDED

PLAINTIFFS' OPPOSITION TO DEFENDANT'S RENEWED MOTION TO DISMISS UNDER RULE 12(B)(6)

TABLE OF CONTENTS

TABLE OF AUTHORITIES

Page(s)

Federal Cases

Rules

I. INTRODUCTION

MARA's Renewed Motion to Dismiss Under Rule 12(b)(6) (Dkt. 38) should be denied. Plaintiffs' (herein, "Malikie") First Amended Complaint ("FAC") includes more than sufficient detail to meet the pleading standard's plausibility requirement. The FAC provides narrative descriptions and annotated source code that together explain in detail where and how MARA infringes element 1[b] of the '286 Patent. But, in order to craft a second motion to dismiss, MARA chooses to ignore much of the relevant detail and mischaracterizes the limited portion that it does consider. MARA's attempt to support its argument with legally erroneous and irrelevant comparisons of the accused instrumentalities to embodiments in the patent's specification cannot support dismissal and instead raises factual disputes and claim construction issues not suitable for resolution on a motion to dismiss. MARA claims that this case is just like *Vervain, LLC v. Micron Tech., Inc.*, No. 6:21-CV-00487-ADA, 2022 WL 23469 (W.D. Tex. Jan. 3, 2022), but overlooks decisive differences distinguishing that case from this one. At bottom, MARA fails to show that Malikie's FAC fails to meet the pleading standard, even for material elements involving complex technology, and its Motion should be denied.

II. APPLICABLE LAW

Fifth Circuit law applies to MARA's Motion to Dismiss. *Alvao Digital, LLC v. C3 Presents, L.L.C.*, No. 6:21-CV-01208-ADA, 2022 WL 22869793, at *2 (W.D. Tex. Sept. 26, 2022) (citing *In re Bill of Lading Transmission & Processing Sys. Pat. Litig.*, 681 F.3d 1323, 1331 (Fed. Cir. 2012)). "A motion to dismiss under rule 12(b)(6) 'is viewed with disfavor and is rarely granted.'" *Collins v. Morgan Stanley Dean Witter*, 224 F.3d 496, 498 (5th Cir. 2000) (quoting *Kaiser Aluminum & Chem. Sales v. Avondale Shipyards*, 677 F.2d 1045, 1050 (5th Cir. 1982)). "The complaint must be liberally construed in favor of the plaintiff, and all facts pleaded in the complaint must be taken as true." *Id.*; *see EcoFactor, Inc v. Ecobee, Inc.*, No. 6:22-CV-00033-

ADA, 2022 WL 3593051, at *2 (W.D. Tex. Aug. 22, 2022) (at the 12(b)(6) stage "the Court 'accepts all well-pleaded facts as true, views them in the light most favorable to the plaintiff, and draws all reasonable inferences in the plaintiff's favor.'" (quoting *Johnson v. BOKF Nat'l Ass'n*, 15 F.4th 356, 361 (5th Cir. 2021))). "To avoid dismissal for failure to state a claim, the complaint must allege 'enough facts to state a claim to relief that is plausible on its face.'" *Carlucci v. Chapa*, 884 F.3d 534, 537-38 (5th Cir. 2018) (quoting *Bell Atl. Corp. v. Twombly*, 550 U.S. 544, 570 (2007)). "A claim has facial plausibility when the plaintiff pleads factual content that allows the court to draw the reasonable inference that the defendant is liable for the misconduct alleged." *Id.* at 538 (quoting *Ashcroft v. Iqbal*, 556 U.S. 662, 678 (2009)); *see Disc Disease Sols. Inc. v. VGH Sols., Inc.*, 888 F.3d 1256, 1260 (Fed. Cir. 2018).

III. THE FIRST AMENDED COMPLAINT SUFFICIENTLY PLEADS INFRINGEMENT OF THE '286 PATENT

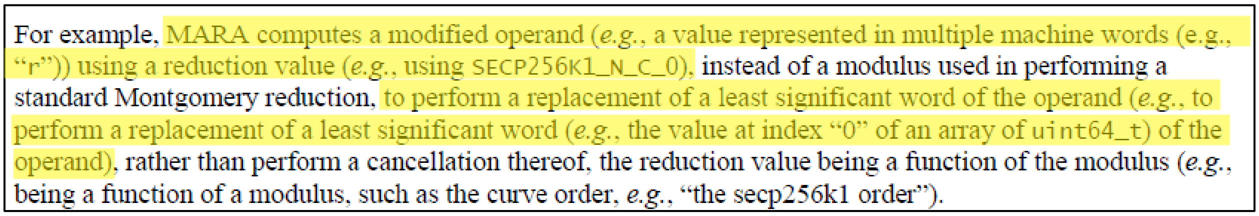
A. Plaintiffs plausibly plead infringement of element 1[b] of the '286 Patent.

Claim 1 of the '286 Patent is a method claim that recites three steps, identified in the FAC as 1[a], 1[b], and 1[c]. MARA's argument is limited to element 1[b] (italicized below), which computes a modified operand using a reduction value to perform a replacement of a least significant word of the operand.

> **[1pre]** A method for performing, on a cryptographic apparatus, a Montgomery-style reduction in a cryptographic operation, the method comprising:
>
> **[1a]** obtaining an operand for the cryptographic operation;
>
> ***[1b]*** *computing a modified operand using a reduction value, instead of a modulus used in performing a standard Montgomery reduction, to perform a replacement of a least significant word of the operand, rather than perform a cancellation thereof, the reduction value being a function of the modulus; and*
>
> **[1c]** outputting the modified operand.

’286 Pat., cl. 1. MARA concedes that “the FAC identifies a ‘reduction value’ and a ‘least significant word of the operand,’” but contends that the FAC does not “identify *where or how* the alleged reduction value is used ‘to perform a replacement of a least significant word of the operand’” and that “Malikie does not plead a logical connection between the code and the Reduction Value Limitation [1b].” Mot. at 9-10 (emphasis added). Both contentions are untrue. The FAC includes a narrative description and annotated source code that together identify where and how the source code practices element 1[b] and provides a logical connection between the source code and element 1[b].[1] MARA’s arguments to the contrary ignore much of the narrative description and source code provided in the FAC.

The FAC explains that “MARA computes a modified operand … (*e.g.*, ‘r’) … using a reduction value (*e.g.*, using SECP256K1_N_C_0) … to perform a replacement of a least significant word (*e.g.*, the value at index “0” of an array of uint64_t) of the operand.” FAC, Ex. 12 at 10.



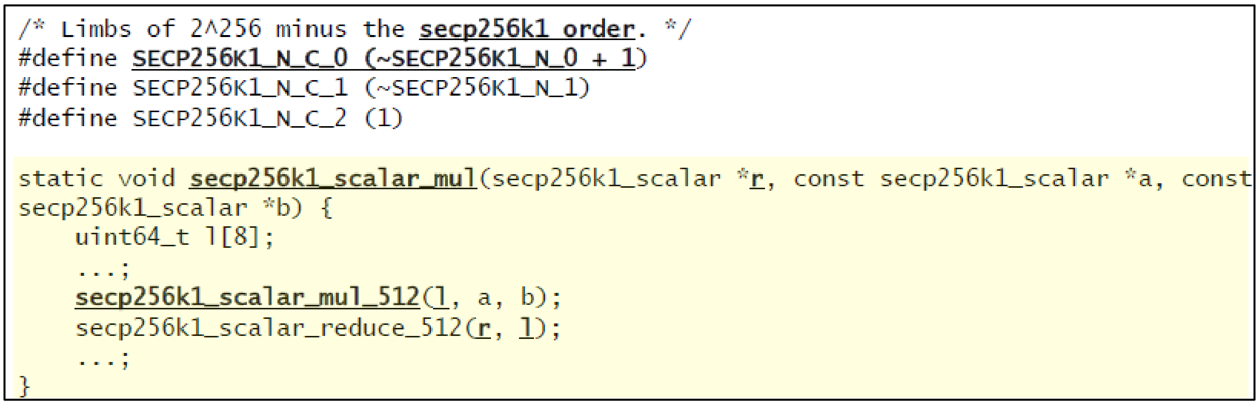
For example, MARA computes a modified operand (*e.g.*, a value represented in multiple machine words (e.g., “r”)) using a reduction value (*e.g.*, using SECP256K1_N_C_0), instead of a modulus used in performing a standard Montgomery reduction, to perform a replacement of a least significant word of the operand (*e.g.*, to perform a replacement of a least significant word (*e.g.*, the value at index “0” of an array of uint64_t) of the operand), rather than perform a cancellation thereof, the reduction value being a function of the modulus (*e.g.*, being a function of a modulus, such as the curve order, *e.g.*, “the secp256k1 order”).

FAC, Ex. 12 at 10

The source code identified in the FAC shows examples of where and how this happens. FAC, Ex. 12 at 10-11. In particular, in the code on page 10 (reproduced below), the function “secp256k1_scalar_mul” generates an output “r” (representing a “modified operand”). A plain reading of the code shows that, to generate “r,” that function defines “l” as an 8-word array of type

[1] Malikie’s allegations at the pleading stage are preliminary. Malikie reserves the right to amend or supplement them as appropriate, based on discovery in this case.

"uint64_t" and invokes the function secp256k1_scalar_mul_512," which generates a value for "l" (representing an "operand"). Then it invokes the function secp256k1_scalar_reduce_512, which takes "l" (representing the "operand") as an input and generates "r" (representing a "modified operand") as an output.



```
/* Limbs of 2^256 minus the secp256k1 order. */
#define SECP256K1_N_C_0 (~SECP256K1_N_0 + 1)
#define SECP256K1_N_C_1 (~SECP256K1_N_1)
#define SECP256K1_N_C_2 (1)

static void secp256k1_scalar_mul(secp256k1_scalar *r, const secp256k1_scalar *a, const
secp256k1_scalar *b) {
    uint64_t l[8];
    ...;
    secp256k1_scalar_mul_512(l, a, b);
    secp256k1_scalar_reduce_512(r, l);
    ...;
}
```

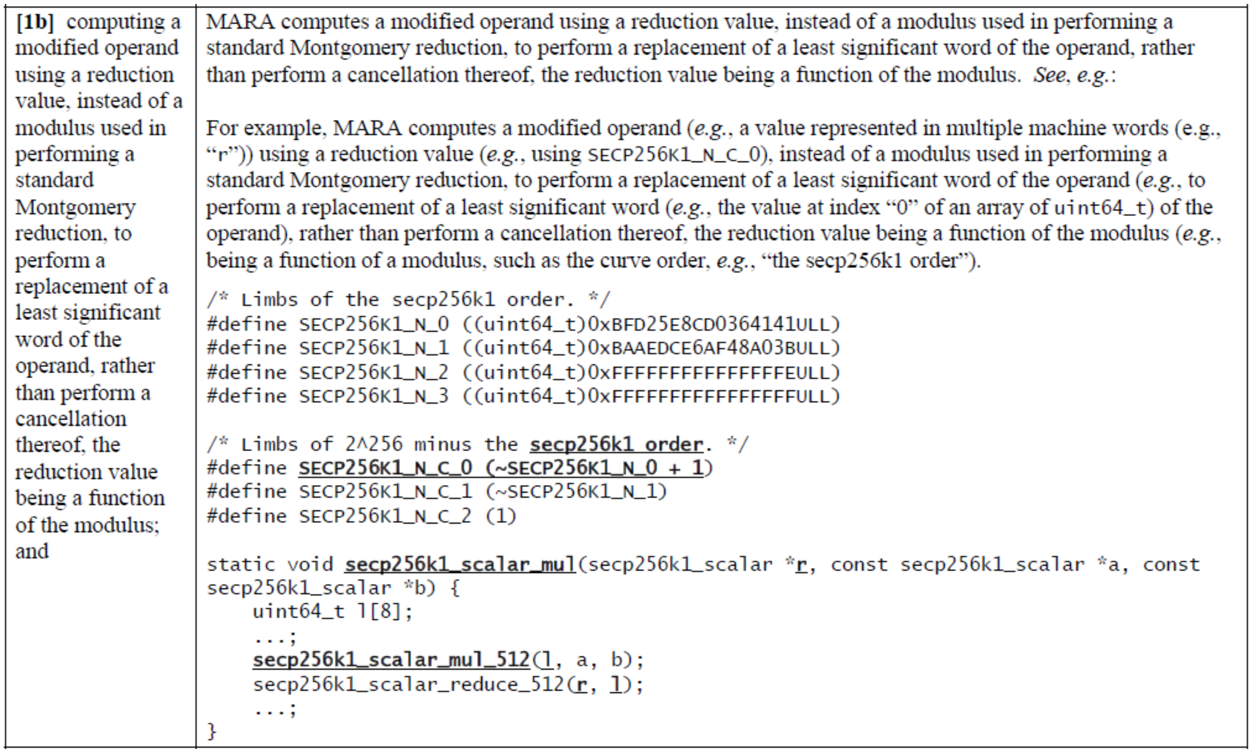
FAC, Ex. 12 at 10 (highlighting added)

Details of the function secp265k1_scalar_reduce_512 are described in the source code (including descriptive "comments"[2]) on page 11 of FAC Exhibit 12 (reproduced below). The comments explain that the machine words of the operand (initially 8 words) are assigned to l[0]–l[3] (the four least significant words) and n[0]–n[3] (the four most significant words)[3]. Then the operand is reduced to 7 words (identified as "m[0 . . 6]"), then to 5 words (identified as "p[0 . . 4]"), and then to 4 words (identified as "r[0 . . 3]"). Below the comments are executable functions

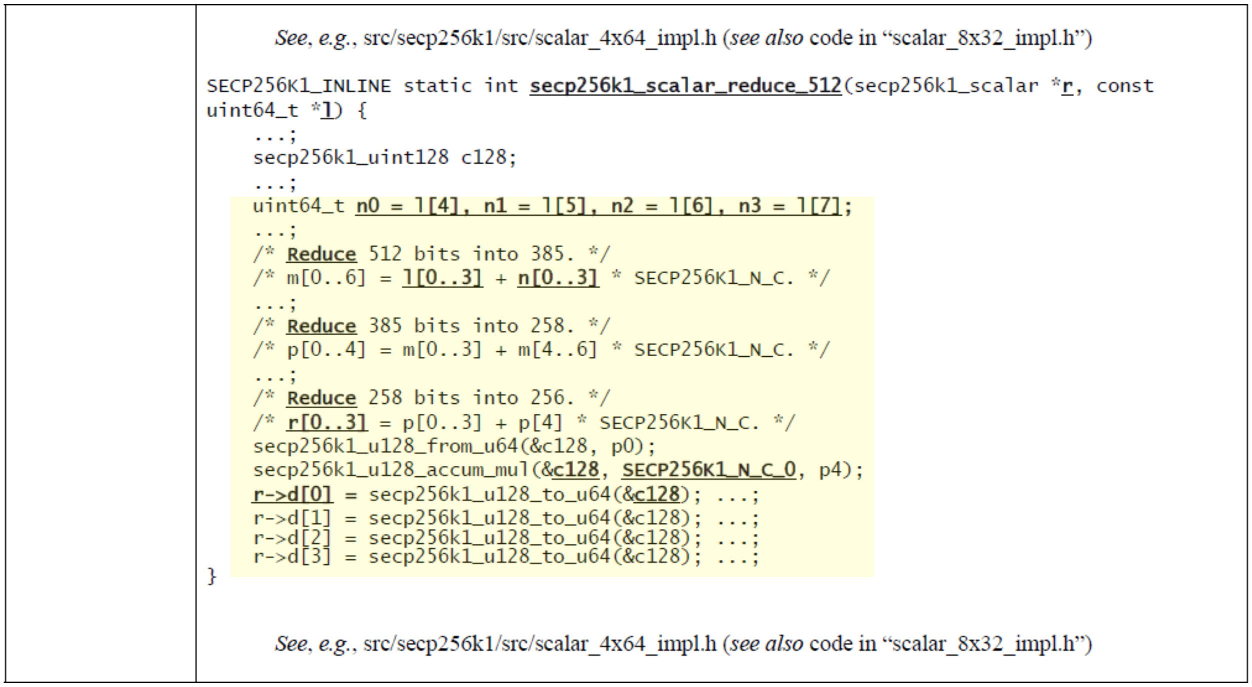
[2] In software development, "comments" are non-executable descriptions embedded in a program by a developer to help human readers understand the purpose of the related code. *See*, *e.g.*, https://www.fda.gov/inspections-compliance-enforcement-and-criminal-investigations/inspection-guides/glossary-computer-system-software-development-terminology-895 (definition of "comments"). Comments are denoted by "/*" at the beginning of each line.

[3] Defendant acknowledges that index value "0" identifies the least significant word. *See* Mot. 10-11. Accordingly, the least significant word of the operand is initially identified by l[0]. As shown in the source code below, n[0] through n[3] hold the values of l[4] through l[7], Accordingly, l[0] through l[3] are initially the four least significant words of the operand, and n[0] through n[3] are initially the four most significant words of the operand. MARA does not dispute this. Mot. 11. As also explained herein, the least significant word of the operand, initially identified as l[0], is subsequently identified as m[0] and then p[0] in the source code.

(actual code) that explain how the final reduction results in a modified operand "r." The function "secp256k1_u128_from_u64" takes the ***least*** significant word of the operand (at that point, identified as "p0") and assigns it to the variable "c128." Then, the function "secp256k1_u128_accum_mul" performs an operation using the input SECP256K1_N_C_0 (a "reduction value"), and it applies the result to the contents of "c128" (*i.e.*, to the least significant word "p0"). Then, each of the words of "r" (a "modified operand") are set one at a time, starting with the least significant word "d[0]" and ending with the most significant word "d[3]." Setting "d[0]" replaces the ***least*** significant word of the operand (what was identified as p[0], and before that as m[0], and before that as l[0]) with the results of "secp256k1_u128_to_u64(&c128)", which uses "c128" as an input, which was generated using SECP256K1_N_C_0 (a "reduction value"). All of this is readily discernable in the specific section of code excerpted and included in the FAC, alongside the corresponding claim element.



[1b] computing a modified operand using a reduction value, instead of a modulus used in performing a standard Montgomery reduction, to perform a replacement of a least significant word of the operand, rather than perform a cancellation thereof, the reduction value being a function of the modulus; and	MARA computes a modified operand using a reduction value, instead of a modulus used in performing a standard Montgomery reduction, to perform a replacement of a least significant word of the operand, rather than perform a cancellation thereof, the reduction value being a function of the modulus. *See, e.g.*:
	For example, MARA computes a modified operand (*e.g.*, a value represented in multiple machine words (e.g., "r")) using a reduction value (*e.g.*, using SECP256K1_N_C_0), instead of a modulus used in performing a standard Montgomery reduction, to perform a replacement of a least significant word of the operand (*e.g.*, to perform a replacement of a least significant word (*e.g.*, the value at index "0" of an array of uint64_t) of the operand), rather than perform a cancellation thereof, the reduction value being a function of the modulus (*e.g.*, being a function of a modulus, such as the curve order, *e.g.*, "the secp256k1 order").
	`/* Limbs of the secp256k1 order. */`
	`#define SECP256K1_N_0 ((uint64_t)0xBFD25E8CD0364141ULL)`
	`#define SECP256K1_N_1 ((uint64_t)0xBAAEDCE6AF48A03BULL)`
	`#define SECP256K1_N_2 ((uint64_t)0xFFFFFFFFFFFFFFFEULL)`
	`#define SECP256K1_N_3 ((uint64_t)0xFFFFFFFFFFFFFFFFULL)`
	
	`/* Limbs of 2^256 minus the secp256k1 order. */`
	`#define SECP256K1_N_C_0 (~SECP256K1_N_0 + 1)`
	`#define SECP256K1_N_C_1 (~SECP256K1_N_1)`
	`#define SECP256K1_N_C_2 (1)`
	
	`static void secp256k1_scalar_mul(secp256k1_scalar *r, const secp256k1_scalar *a, const secp256k1_scalar *b) {`
	`    uint64_t l[8];`
	`    ...;`
	`    secp256k1_scalar_mul_512(l, a, b);`
	`    secp256k1_scalar_reduce_512(r, l);`
	`    ...;`
	`}`



See, *e.g.*, src/secp256k1/src/scalar_4x64_impl.h (*see also* code in "scalar_8x32_impl.h")

```
SECP256K1_INLINE static int secp256k1_scalar_reduce_512(secp256k1_scalar *r, const
uint64_t *l) {
    ...;
    secp256k1_uint128 c128;
    ...;
    uint64_t n0 = l[4], n1 = l[5], n2 = l[6], n3 = l[7];
    ...;
    /* Reduce 512 bits into 385. */
    /* m[0..6] = l[0..3] + n[0..3] * SECP256K1_N_C. */
    ...;
    /* Reduce 385 bits into 258. */
    /* p[0..4] = m[0..3] + m[4..6] * SECP256K1_N_C. */
    ...;
    /* Reduce 258 bits into 256. */
    /* r[0..3] = p[0..3] + p[4] * SECP256K1_N_C. */
    secp256k1_u128_from_u64(&c128, p0);
    secp256k1_u128_accum_mul(&c128, SECP256K1_N_C_0, p4);
    r->d[0] = secp256k1_u128_to_u64(&c128); ...;
    r->d[1] = secp256k1_u128_to_u64(&c128); ...;
    r->d[2] = secp256k1_u128_to_u64(&c128); ...;
    r->d[3] = secp256k1_u128_to_u64(&c128); ...;
}
```

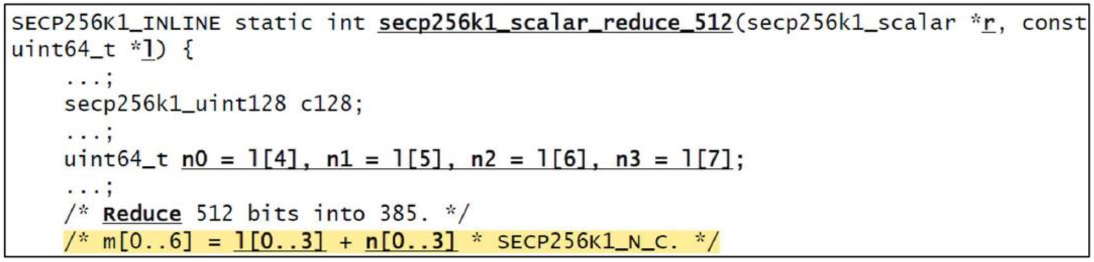
See, *e.g.*, src/secp256k1/src/scalar_4x64_impl.h (*see also* code in "scalar_8x32_impl.h")

FAC, Ex. 12 at 10-11 (highlighting added)

The FAC thus identifies with specificity where and how MARA's use of the cited code practices element 1[b] and provides a logical connection between the cited code and element 1[b].[4]

MARA's separate attacks on Malikie's narrative descriptions ignore the fact that the descriptions and annotated source code, when considered together, provide a clear logical connection between element 1[b] and the code by, for example, indicating where (through exemplary variables and functions) and how (by executing the code corresponding to those functions) element 1[b] is plausibly infringed. Indeed, MARA's argument skips over the narrative explanation's identification of a modified operand (*e.g.*, "r") and much of the source code related to computing it. In particular, MARA's motion ignores (without explanation) all of the code on page 11 of FAC Exhibit 12 below the portion excerpted in its Motion—code that the FAC uses to plausibly plead infringement of element 1[b]. *Compare* Mot. 11's excerpt

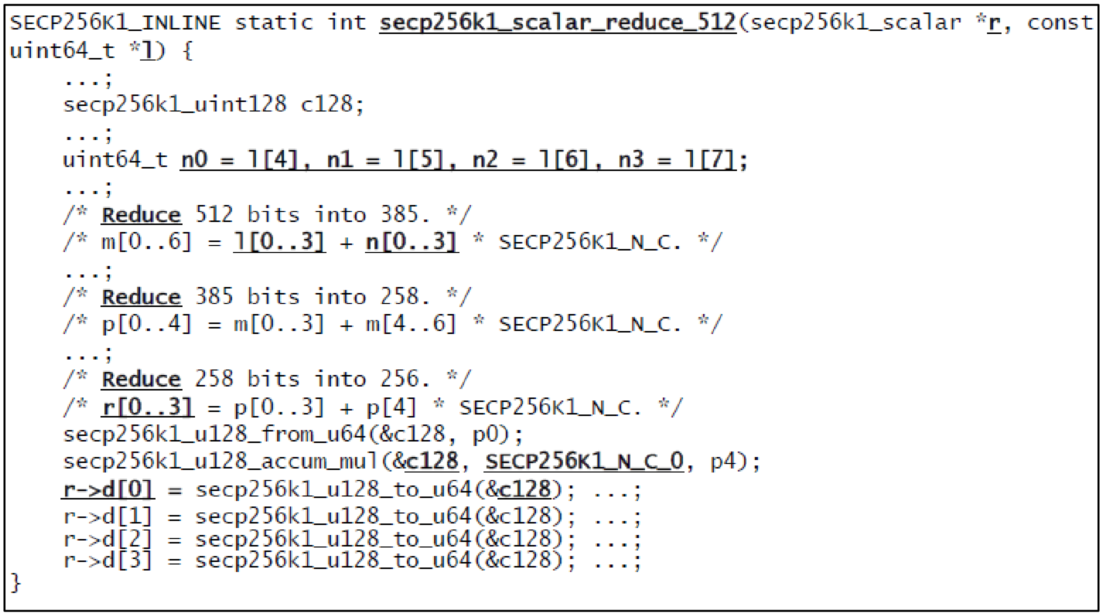
[4] MARA's motion does not dispute that MARA uses the code cited in the FAC. The cited code is from the Bitcoin Core open source project. *See* Dkt. 36 ¶22. MARA has publicly acknowledged that it uses Bitcoin Core software. Dkt. 36-12 at 3-4 (citing MARA press release).



```
SECP256K1_INLINE static int secp256k1_scalar_reduce_512(secp256k1_scalar *r, const
uint64_t *l) {
    ...;
    secp256k1_uint128 c128;
    ...;
    uint64_t n0 = l[4], n1 = l[5], n2 = l[6], n3 = l[7];
    ...;
    /* Reduce 512 bits into 385. */
    /* m[0..6] = l[0..3] + n[0..3] * SECP256K1_N_C. */
```

FAC, Ex. 12 at 11 (annotated)

with the fulsome excerpt in the FAC, Ex. 12 at 11

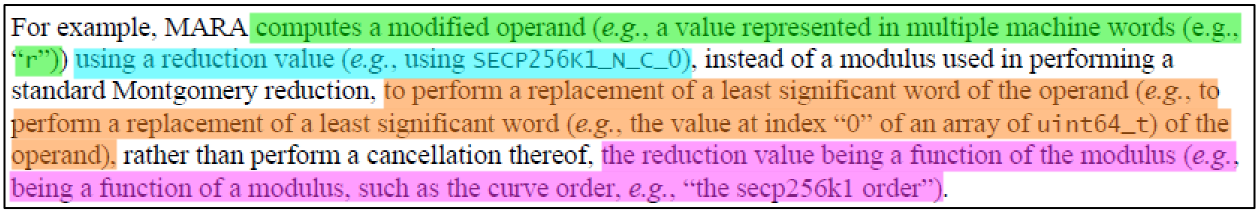


```
SECP256K1_INLINE static int secp256k1_scalar_reduce_512(secp256k1_scalar *r, const
uint64_t *l) {
    ...;
    secp256k1_uint128 c128;
    ...;
    uint64_t n0 = l[4], n1 = l[5], n2 = l[6], n3 = l[7];
    ...;
    /* Reduce 512 bits into 385. */
    /* m[0..6] = l[0..3] + n[0..3] * SECP256K1_N_C. */
    ...;
    /* Reduce 385 bits into 258. */
    /* p[0..4] = m[0..3] + m[4..6] * SECP256K1_N_C. */
    ...;
    /* Reduce 258 bits into 256. */
    /* r[0..3] = p[0..3] + p[4] * SECP256K1_N_C. */
    secp256k1_u128_from_u64(&c128, p0);
    secp256k1_u128_accum_mul(&c128, SECP256K1_N_C_0, p4);
    r->d[0] = secp256k1_u128_to_u64(&c128); ...;
    r->d[1] = secp256k1_u128_to_u64(&c128); ...;
    r->d[2] = secp256k1_u128_to_u64(&c128); ...;
    r->d[3] = secp256k1_u128_to_u64(&c128); ...;
}
```

.

MARA's failure to address the detailed evidence included in the FAC discredits its argument that the FAC lacks sufficient detail.

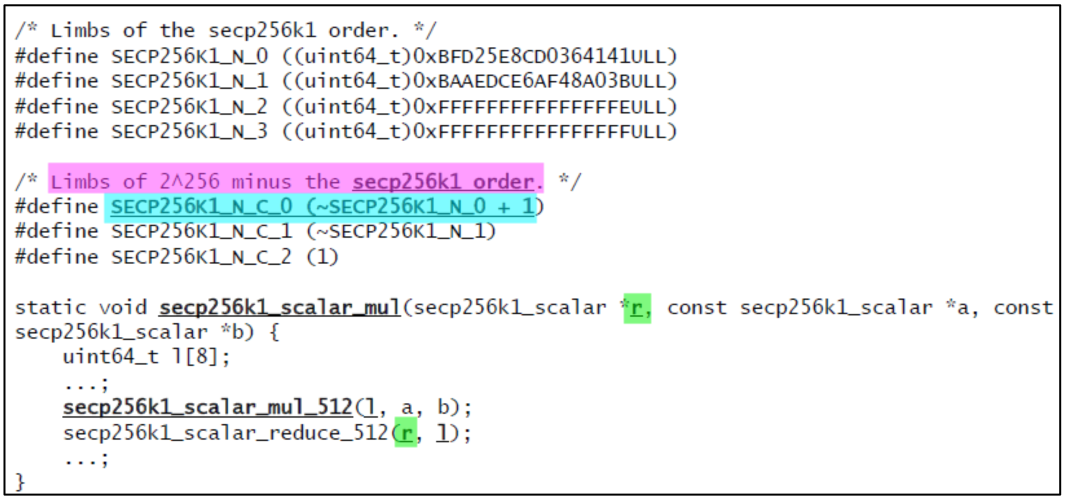
MARA's argument that "the relevance of the bold code is unclear and unexplained" is belied by the evidence. Mot. 10. As MARA itself admits, the bolded code in the FAC directs MARA to variables that are relevant to element 1[b]. *Id.* at 9 ("The second statement identifies accused ***variables*** in the source code ....") (emphasis in original). The narrative goes on to explain how those variables relate to element 1[b]:



For example, MARA computes a modified operand (*e.g.*, a value represented in multiple machine words (e.g., "r")) using a reduction value (*e.g.*, using `SECP256K1_N_C_0`), instead of a modulus used in performing a standard Montgomery reduction, to perform a replacement of a least significant word of the operand (*e.g.*, to perform a replacement of a least significant word (*e.g.*, the value at index "0" of an array of `uint64_t`) of the operand), rather than perform a cancellation thereof, the reduction value being a function of the modulus (*e.g.*, being a function of a modulus, such as the curve order, *e.g.*, "the secp256k1 order").

FAC, Ex. 12 at 10 (colors added)

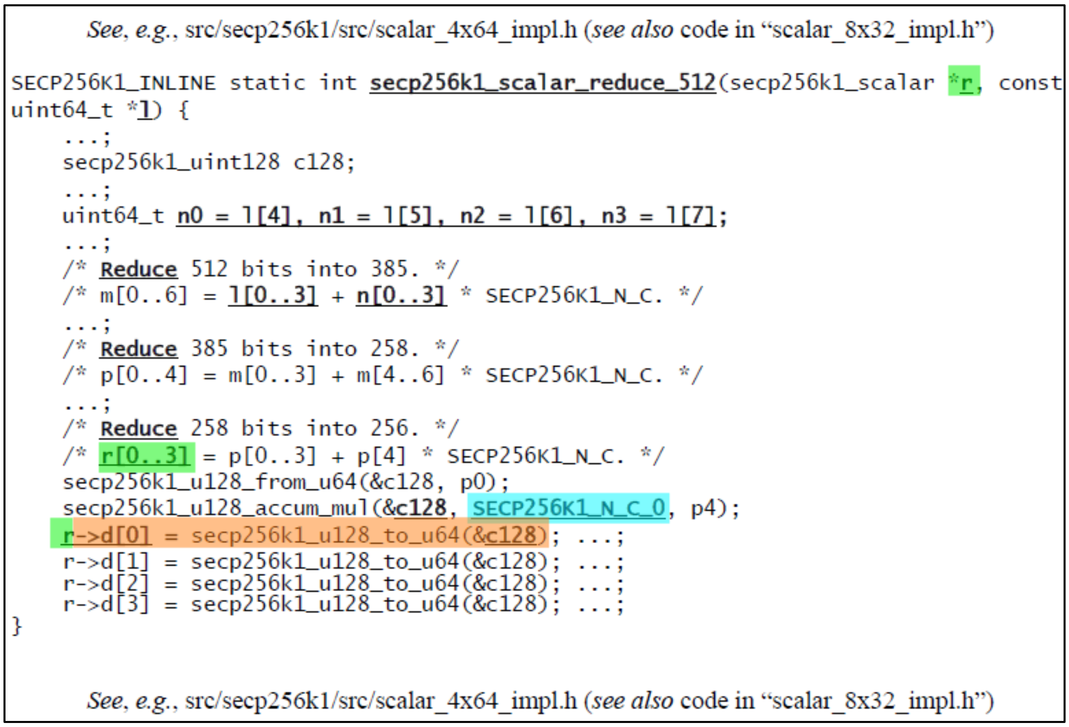
The provision of annotated source code in the FAC shows where and how those variables are used to perform element 1[b] by identifying the sequence of functions that operate on those variables.



```
/* Limbs of the secp256k1 order. */
#define SECP256K1_N_0 ((uint64_t)0xBFD25E8CD0364141ULL)
#define SECP256K1_N_1 ((uint64_t)0xBAAEDCE6AF48A03BULL)
#define SECP256K1_N_2 ((uint64_t)0xFFFFFFFFFFFFFFFEULL)
#define SECP256K1_N_3 ((uint64_t)0xFFFFFFFFFFFFFFFFULL)

/* Limbs of 2^256 minus the secp256k1 order. */
#define SECP256K1_N_C_0 (~SECP256K1_N_0 + 1)
#define SECP256K1_N_C_1 (~SECP256K1_N_1)
#define SECP256K1_N_C_2 (1)

static void secp256k1_scalar_mul(secp256k1_scalar *r, const secp256k1_scalar *a, const
secp256k1_scalar *b) {
    uint64_t l[8];
    ...;
    secp256k1_scalar_mul_512(l, a, b);
    secp256k1_scalar_reduce_512(r, l);
    ...;
}
```



See, *e.g.*, src/secp256k1/src/scalar_4x64_impl.h (*see also* code in "scalar_8x32_impl.h")

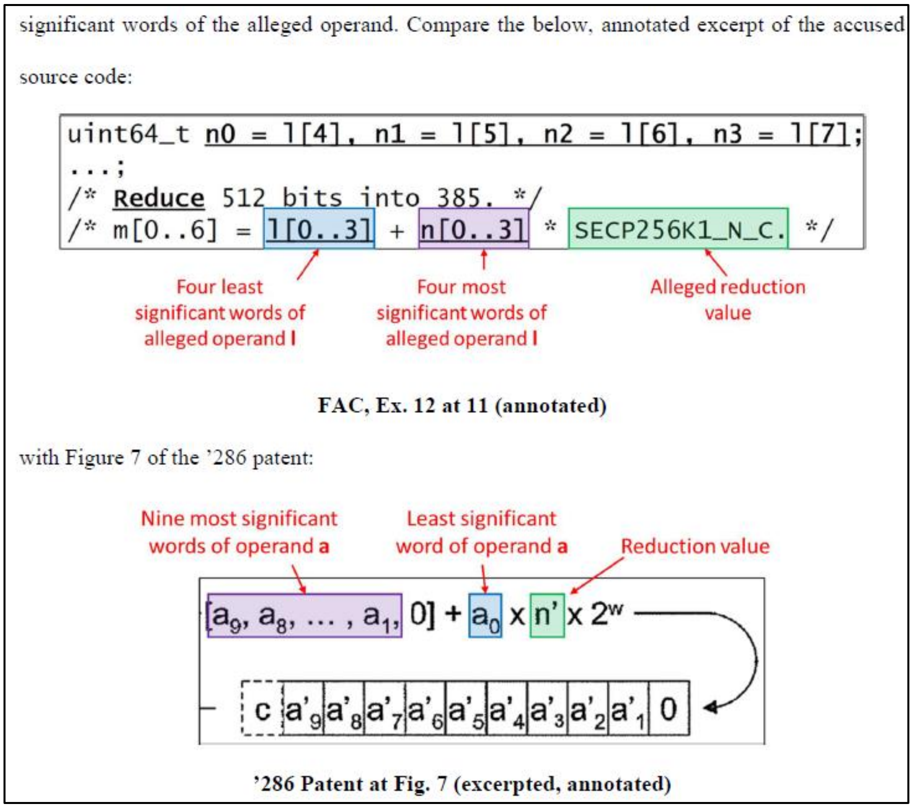
```
SECP256K1_INLINE static int secp256k1_scalar_reduce_512(secp256k1_scalar *r, const
uint64_t *l) {
    ...;
    secp256k1_uint128 c128;
    ...;
    uint64_t n0 = l[4], n1 = l[5], n2 = l[6], n3 = l[7];
    ...;
    /* Reduce 512 bits into 385. */
    /* m[0..6] = l[0..3] + n[0..3] * SECP256K1_N_C. */
    ...;
    /* Reduce 385 bits into 258. */
    /* p[0..4] = m[0..3] + m[4..6] * SECP256K1_N_C. */
    ...;
    /* Reduce 258 bits into 256. */
    /* r[0..3] = p[0..3] + p[4] * SECP256K1_N_C. */
    secp256k1_u128_from_u64(&c128, p0);
    secp256k1_u128_accum_mul(&c128, SECP256K1_N_C_0, p4);
    r->d[0] = secp256k1_u128_to_u64(&c128); ...;
    r->d[1] = secp256k1_u128_to_u64(&c128); ...;
    r->d[2] = secp256k1_u128_to_u64(&c128); ...;
    r->d[3] = secp256k1_u128_to_u64(&c128); ...;
}
```

See, *e.g.*, src/secp256k1/src/scalar_4x64_impl.h (*see also* code in "scalar_8x32_impl.h")

FAC, Ex. 12 at 11-12 (colors added)

The narrative descriptions and annotated source code in the FAC thus provide sufficient detail to plausibly plead infringement, particularly considering MARA's demonstrated ability to understand and interpret the source code (evidenced by its own source code analysis, Mot. 10-12). *See Castlemorton Wireless, LLC v. Bose Corp.*, No. 6:20-CV-00029-ADA, 2020 WL 6578418, at *4 (W.D. Tex. July 22, 2020) ("[Plaintiff] alleges specific facts as to precisely how the accused products utilize [the claimed] method … These are hardly 'mere conclusory statements' that fail to plausibly allege [the defendant] is liable for infringement of the [asserted] patent, and the Court accordingly denies [the defendant's] motion to dismiss.").

MARA's argument that none of the source code cited in the FAC "bears any resemblance to the 'reduction value' described and claimed in the '286 patent" is also wrong. The FAC explains in detail how element 1[b] maps to the cited code, as explained above. Moreover, the alleged lack of resemblance MARA complains of is the result of MARA's mischaracterization of the replacement process in the code cited in the FAC and its improper comparison of the accused instrumentalities to embodiments in the '286 Patent's specification. MARA argues that the source code "shows the alleged reduction value being multiplied with the ***most*** significant words of the alleged operand," and for that reason argues that "the alleged 'reduction value' SECP256K1_N_C is used to <u>perform a replacement of the ***most significant*** words</u> of the alleged operand **l**" rather than the least significant word as claimed. Mot. 11-12 (underlining added); *id*. at 1 (asserting that "the quoted source code shows <u>replacement of the ***most*** significant words</u> of the operand, which is the ***opposite*** of what is claimed" (underlining added)). In light of this alleged inconsistency, MARA asserts that "Malikie has no basis to allege that MARA practices this limitation." *Id*. 12. MARA supports its argument by comparing only a portion of one line of the source code comments cited in the FAC to an embodiment in the '286 Patent's specification:



significant words of the alleged operand. Compare the below, annotated excerpt of the accused source code:

```
uint64_t n0 = l[4], n1 = l[5], n2 = l[6], n3 = l[7];
...;
/* Reduce 512 bits into 385. */
/* m[0..6] = l[0..3] + n[0..3] * SECP256K1_N_C. */
```

Four least significant words of alleged operand l
Four most significant words of alleged operand l
Alleged reduction value

FAC, Ex. 12 at 11 (annotated)

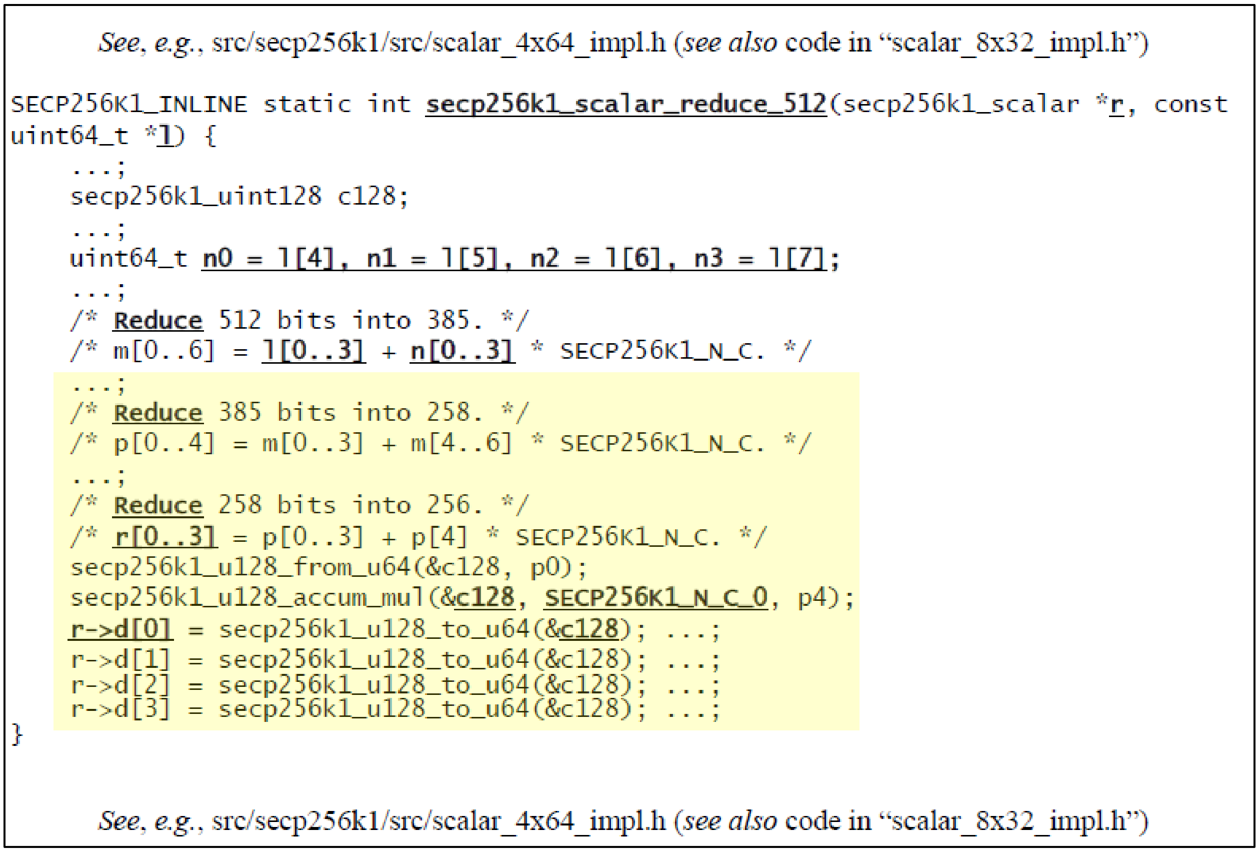
with Figure 7 of the '286 patent:

Nine most significant words of operand a
Least significant word of operand a
Reduction value
[a_9, a_8, … , a_1, 0] + a_0 x n' x 2^w
c a'_9 a'_8 a'_7 a'_6 a'_5 a'_4 a'_3 a'_2 a'_1 0

'286 Patent at Fig. 7 (excerpted, annotated)

Mot. 12

MARA's argument fails for multiple reasons. First, it fails to support its assumption that the *multiplication* it points to is a "*replacement*" of the most significant word of the operand. As explained above, the FAC alleges that the replacement of the least significant word of the operand (*e.g.*, replacement of "the value at index '0' of an array of uint64_t") for computing the modified operand (*e.g.*, "r") involves operations for setting the value at index "0" to the result of an operation that uses the reduction value "SECP256K1_N_C_0." *See supra*. The FAC thus shows where and how the least significant word of the operand—not the most significant word—is replaced. Accordingly, there are no "irreconcilable internal inconsistencies" in the FAC, as MARA suggests, Mot. 12 (citing *Qwikcash, LLC v. Blackhawk Network Holdings, Inc.*, No. 4:19-CV-876-SDJ, 2020 WL 6781566, at *4–5 (E.D. Tex. Nov. 17, 2020)). Rather, MARA simply ignored the portions of the annotated source code that show how the preplacement is performed (highlighted below), which dovetails with FAC's companying narrative explanation.



See, e.g., src/secp256k1/src/scalar_4x64_impl.h (*see also* code in "scalar_8x32_impl.h")

```
SECP256K1_INLINE static int secp256k1_scalar_reduce_512(secp256k1_scalar *r, const
uint64_t *l) {
    ...;
    secp256k1_uint128 c128;
    ...;
    uint64_t n0 = l[4], n1 = l[5], n2 = l[6], n3 = l[7];
    ...;
    /* Reduce 512 bits into 385. */
    /* m[0..6] = l[0..3] + n[0..3] * SECP256K1_N_C. */
    ...;
    /* Reduce 385 bits into 258. */
    /* p[0..4] = m[0..3] + m[4..6] * SECP256K1_N_C. */
    ...;
    /* Reduce 258 bits into 256. */
    /* r[0..3] = p[0..3] + p[4] * SECP256K1_N_C. */
    secp256k1_u128_from_u64(&c128, p0);
    secp256k1_u128_accum_mul(&c128, SECP256K1_N_C_0, p4);
    r->d[0] = secp256k1_u128_to_u64(&c128); ...;
    r->d[1] = secp256k1_u128_to_u64(&c128); ...;
    r->d[2] = secp256k1_u128_to_u64(&c128); ...;
    r->d[3] = secp256k1_u128_to_u64(&c128); ...;
}
```

See, e.g., src/secp256k1/src/scalar_4x64_impl.h (*see also* code in "scalar_8x32_impl.h")

FAC, Ex. 12 at 11 (highlighting added)

The alleged lack of resemblance and inconsistencies on which MARA bases its arguments are thus the result of its own failure to consider *all* of cited code in context of the *full* narrative description provided in the FAC, not any deficiency in the FAC itself. At the very least, MARA's arguments reveal a dispute of fact as to how the source code works, which precludes dismissal. *Sensor Elec. Tech., Inc. v. Lite-On Tech. Corp.*, No. 6:21-CV-322-ADA, 2022 WL 22879658, at *5 (W.D. Tex. Aug. 22, 2022) ("[T]he purported contradiction in SETi's FAC is a factual dispute not meant for resolution at the pleading stage."); *Unification Techs. LLC v. Dell Techs., Inc.*, No. 6:20-CV-00499-ADA, 2021 WL 1343188, at *3 (W.D. Tex. Jan. 28, 2021) ("[A] 12(b)(6) motion is not the appropriate procedure for identifying inconsistent direct infringement contentions. Those are premature assertions that are best addressed in claim construction or non-infringement positions."); *see also Smith v. Reg'l Transit Auth.*, 756 F.3d 340, 347 (5th Cir. 2014) ("disputed questions of fact are anathema to Rule 12(b)(6) jurisprudence"); *Lone Star Fund V (U.S.), L.P. v. Barclays Bank PLC*, 594 F.3d 383, 387 (5th Cir. 2010) ("The court's task is to determine whether the plaintiff has stated a legally cognizable claim that is plausible, not to evaluate the plaintiff's likelihood of success." (*citing Iqbal*, 556 U.S. at 678)).

Second, MARA's comparison of the code cited in the FAC to the figures of the '286 Patent's specification is legally erroneous and improper. It is well established that "infringement is to be determined by comparing the asserted claim to the accused device, not by comparing the accused device to the figures of the asserted patent." *Catalina Lighting, Inc. v. Lamps Plus, Inc.*, 295 F.3d 1277, 1286 (Fed. Cir. 2002). MARA's comparison argument is therefore irrelevant.

Finally, MARA's deference to the specification and figures to interpret the scope of the claims is a matter of claim construction that further warrants denial of its Motion. *Fintiv, Inc. v. PayPal Holdings, Inc.*, No. 6:22-CV-288-ADA, 2022 WL 22870185, at *5 (W.D. Tex. Dec. 19, 2022) ("Resolution of disputes regarding construction of claims is not appropriate at the motion to dismiss stage."); *id* at *9 ("The Court agrees and finds that PayPal's arguments here hinge on the construction of the 'agent' limitation. As the Court explained above, this again is improper at this stage of the case."); *Micron*, 2022 WL 23469 at *7 ("Micron's arguments … draw too-thin distinctions better suited to resolution at a later stage, after the claims have been construed.").

For all the reasons above, MARA's arguments that Malikie has not adequately alleged (and that it has no basis to allege) infringement because the code cited in the FAC is not (in its opinion) similar to embodiments in the patent's specification should be rejected. In all events, the code cited in the FAC is similar to embodiments described in the specification because the cited code, like the embodiments in the patent, replaces a least significant word of the operand using a modified operand, as explained above.

B. Defendant's Reliance on *Micron* is Misplaced.

MARA argues that it should prevail "for the same reasons this Court articulated in *Vervain, LLC v. Micron Tech., Inc.*, No. 6:21-CV-00487-ADA, 2022 WL 23469, at *5 (W.D. Tex. Jan. 3, 2022) (Albright, J.)." Mot. at 1. However, the only purported similarities that MARA identifies between this case and *Micron* are that the technology is "not simple" and the claim elements at issue are "material." *Id.* at 8-9. Taking that as true, MARA still fails to show that *Micron* is analogous, such that the rationale of *Micron* should apply here. To the contrary, this case is distinguishable because, unlike in *Micron*, Malikie's FAC includes sufficient detail to show why it is plausible that MARA infringes the alleged "point of novelty," *i.e.*, element 1[b]. *See Micron*, 2022 WL 23469 at *5.

In *Micron*, the claim at issue required a controller to perform a number of steps, including two that were referred to as the "Hot Blocks Limitations." *Micron*, 2022 WL 23469 at *5. The Hot Blocks Limitations required "transferring the respective contents of [the blocks determined to be accessed most frequently] to the at least one SLC non-volatile memory module." *Id.* at *4. The complaint alleged that the accused products use "wear-leveling" processes to practice the Hot Blocks Limitations and cited product documentation describing the wear-leveling process. *Id.* Micron argued that because the product documentation described the wear-leveling processes as "entirely agnostic as to the type of memory … in which a data block is stored," it did not show that the Hot Blocks Limitations were satisfied. *Id.* In response, Vervain argued that the product documentation "never says wear leveling does not write … to SLC blocks." *Id.* Given this question of whether the wear-leveling processes in the accused products transfer contents from the most frequently accessed blocks *to SLC blocks*, the Court held that "[t]he level of detail provided in Vervain's allegations did not meet the standard here, where the technology is not simple and the limitations-at-issue are material." *Id.* at *5.[5] The need for more detailed pleadings, however, arose not just from the complexity of the technology and materiality of the limitations, but from the Court's determination that the accused wear-leveling processes were not shown to be anything more than conventional wear-leveling processes—*i.e.*, they were not shown to plausibly include the novel steps of the Hot Blocks Limitations (transfer to SLC blocks). Rather, the Court determined that the product documentation's description of the accused wear-leveling processes was "consistent with the [asserted] patent's own description, in the 'BACKGROUND OF THE DISCLOSURE' section, of the same two wear-leveling techniques." *Id.* at *6. "***In this context***, the Court refuse[d] to find that Micron's ***practicing techniques the Asserted Patents characterize as conventional***—wear leveling—leads to a reasonable inference that Micron practices the presumably novel invention claimed in these three Asserted Patents." *Id.* (emphasis added).

In contrast to the situation in *Micron*, MARA does not argue (and raises no evidence in its Motion to suggest) that the source code cited in the FAC practices only a conventional reduction process rendering the FAC lacking in detail. Indeed, MARA's motion concedes that the invention here is "different[]" from what the '286 Patent's specification identified as prior art. Mot. 8. Nor does MARA provide any other reason as to *why* additional details would be needed—even assuming element 1[b] is material and complex—such that the FAC does not meet the pleading standard's plausibility requirement. Accordingly, this case is distinguishable from *Micron*.

MARA's materiality arguments do not explain why the FAC lacks sufficient detail to plausibly plead infringement. MARA alleges that the "sole difference" between the invention here

[5] The Court held that the Hot Blocks Limitations were material because they "lay at the point of novelty," as informed by patent's prosecution history. *Id.* at *5.

and the prior art "is in the use of a particular 'reduction value,'" which MARA claims is confirmed by the prosecution history.[6] Mot. 8-9. But MARA never argues that the source code cited in the FAC uses only a conventional reduction value in a conventional way. Nor does MARA dispute that the claimed reduction value is present in the code. Indeed, MARA acknowledges that the FAC identifies a reduction value. Mot. 11 ("SECP256K1_N_C is the alleged 'reduction value'."). MARA never argues that SECP256K1_N_C is a conventional reduction value or that it is used in a conventional way. MARA thus fails to show that the reasoning in *Micron* is applicable here.

MARA also fails to demonstrate that the FAC lacks sufficient detail due to the complexity of the technology at issue. MARA asserts that "[t]he level of detail provided in the FAC does not meet the pleading standard here, '*where the technology is not simple* and the limitations-at-issue are material.'" Mot. 9 (quoting *Micron*, at *5) (emphasis added). But MARA does not explain why the complexity of the technology at issue renders the FAC insufficient. The FAC cites detailed source code that shows where and how the accused instrumentalities practice the entirety of element 1[b], including the "reduction value" and "replacement" limitations. *See supra*. This is more detailed evidence than the product documentation in *Micron*, which did *not* explain how the accused products worked with enough specificity to plausibly show that the Hot Blocks Limitations were satisfied. *See Micron*, 2022 WL 23469 at *4 ("Vervain's opposition remarks how the WL Note 'never says wear leveling does not write from MLC blocks to SLC blocks.'").

[6] The prosecution history shows that the inventive technique is more than just the use of a "reduction value," but also how it is used to compute a modified operand according to the claims. *See* '286 Pat., cl. 1. In the Notice of Allowance, the examiner indicated that "the prior art fails to teach or reasonably suggest the invention ***as claimed***," and for its reasoning directed the reader to "the applicant's remarks filed on 5/1/13, which contrasts the cited references against the invention ***as claimed***." Mot. 8-9 (citing Mot. Ex. 3 at 2) (emphasis added). Those remarks indicate that the invention includes the use of a reduction value ***and*** as how it is used to perform reduction differently: "[N]ot only is the claimed method used to ***perform a reduction differently***, a reduction value is used." Mot. 8 (citing Mot. Ex. 2 at 7) (emphasis added).

MARA's argument that Malikie's detailed FAC fails to meet the pleading standard "demand[s] too much." *Micron*, 2022 WL 23469 at *2 (discussing *Bot M8 LLC v. Sony Corporation of America*, 4 F.4th 1342, 1356 (Fed. Cir. 2021)). Malikie "need not 'prove its case at the pleading stage.'" *Bot M8*, 4 F.4th at 1356 (quoting *Nalco Co. v. Chem-Mod, LLC*, 883 F.3d 1337, 1350 (Fed. Cir. 2018)). Because MARA does not articulate a valid reason *why* the FAC lacks sufficient detail, the Court should deny its Motion.

C. The Court Should Grant Leave to Amend.

Although Malikie believes its FAC plausibly pleads infringement of the '286 Patent, were the Court to find otherwise, Malikie respectfully requests leave to amend to include additional details. "[A] party's request to amend pleadings should be granted in the absence of 'undue delay, bad faith or dilatory motive on the part of the movant, repeated failure to cure deficiencies by amendments previously allowed, undue prejudice to the opposing party, and futility of amendment.'" *Alvao Digital, LLC v. C3 Presents, L.L.C.*, No. 6:21-CV-01208-ADA, 2022 WL 22869793, at *3 (W.D. Tex. Sept. 26, 2022) (quoting *Foman v. Davis*, 371 U.S. 178, 182 (1962)); *see Dussouy v. Gulf Coast Inv. Corp.*, 660 F.2d 594, 597-98 (5th Cir. 1981).

Here, none of the mitigating factors favor denying leave to amend. Malikie did not delay in filing its FAC just three days after MARA's first motion to dismiss,[7] Dkts. 17, 35. *See Alvao Digital*, 2022 WL 22869793 at *3 ("Alvao additionally has only amended pleadings once thus far, and therefore has not yet expended the Court's patience." (citing *Bennett v. McDermott Int'l., Inc.*, 855 Fed.App'x 932, 940 (5th Cir. 2021))). The FAC was a good faith attempt to cure the alleged deficiencies in the original Complaint, *compare* Dkt. 1, Ex. 12 at 7-9 *with* Dkt. 36, Ex. 12 at 10-

[7] Malikie refiled its FAC on July 25, 2025 (four days after MARA's first motion to dismiss), Dkt. 36, to correct a typographical error in the header.

11 (adding details to the analysis of element 1[b]), and no bad faith conduct been alleged (as there has been none). There is no prejudice to MARA because this case is still in a very early stage and the proposed amendment would "not radically alter the nature of the case." *Alvao Digital*, 2022 WL 22869793 at *3 (granting leave to amend). The proposed amendment would include more definite statements of fact relating to the correspondence of element 1[b] to the source code cited in the FAC. *See id*. ("The specific statements being altered or added will undoubtedly be tailored to what the Court identifies as deficient in this Order, and therefore the lack of further detail is not of great concern."). Thus, while Malikie believes the FAC plausibly pleads infringement of the '286 Patent, were the Court to find otherwise, the Court should grant Malikie leave to amend.

IV. CONCLUSION

Defendant's Motion should be denied in its entirety; however, if the Court finds otherwise, leave to amend should be granted for the reasons set forth above.

Dated: August 18, 2025

Respectfully Submitted,

/s/ Khue V. Hoang
Khue V. Hoang
REICHMAN JORGENSEN
LEHMAN & FELDBERG LLP
400 Madison Avenue, Suite 14D
New York, NY 10017
Tel: (212) 381-1965
khoang@reichmanjorgensen.com

Matthew G. Berkowitz
REICHMAN JORGENSEN
LEHMAN & FELDBERG LLP
100 Marine Parkway, Suite 300
Redwood Shores, CA 94065
Tel: (650) 623-1401
mberkowitz@reichmanjorgensen.com

Philip J. Eklem
REICHMAN JORGENSEN
LEHMAN & FELDBERG LLP
1909 K Street NW, Suite 800
Washington DC, 20006
Tel: (202) 894-7310
peklem@reichmanjorgensen.com

Of Counsel:

Mark D. Siegmund, TX Bar No. 24117055
Cherry Johnson Siegmund James, PC
7901 Fish Pond Rd., 2nd Floor
Waco, TX 76710
Telephone: (254) 732-2242
Facsimile: (866) 627-3509
Email: msiegmund@cjsjlaw.com

Attorneys for Plaintiffs
Malikie Innovations, Ltd. and
Key Patent Innovations, Ltd.

CERTIFICATE OF SERVICE

I hereby certify that all counsel of record are being served with a copy of the foregoing Plaintiffs' Opposition to Defendant's Renewed Motion to Dismiss Under Rule 12(b)(6) via the Court's CM/ECF system on August 18, 2025.

/s/ Mark D. Siegmund
Mark D. Siegmund